/\*

2.Create a SQL store process to register the use of the database, complete it with proper validation and

transaction rollback and commit.

\*/

create procedure spupdatesalary

as

begin

begin try

begin tran

update [dbo].[tblemployee] set salary = 9000 where id = 1

commit tran

end try

begin catch

rollback tran

end catch

end

exec spupdatesalary

/\*3. List the SQL aggregate function and demonstrate how to utilize it.

o SQL aggregation function is used to perform the calculations on multiple rows of a single column of a table. It returns a single value. It is also used to summarize the data.

o Types of SQL Aggregation Function

1. COUNT FUNCTION

2. SUM Function

3. AVG function

4. MAX Function

5. MIN Function

COUNT FUNCTION: COUNT function is used to Count the number of rows in a database table. It can work on both numeric and non-numeric data types. COUNT function uses the COUNT (\*) that returns the count of all the rows in a specified table. COUNT (\*) considers duplicate and Null.

Ex: select count(\*) from employees ------returns no of rows in employees table

SUM Function: Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.

Ex: select sum(marks) from student --------------returns the sum of marks column in student table.

AVG function: The AVG function is used to calculate the average value of the numeric type. AVG function returns the average of all non-Null values.

Ex: select avg(marks) from student ----------returns average of marks column in student table.

MAX Function: MAX function is used to find the maximum value of a certain column. This function determines the largest value of all selected values of a column.

Ex: select max(marks) from student ----------returns maximum value from marks column in student table.

MIN Function: MIN function is used to find the minimum value of a certain column. This function determines the smallest value of all selected values of a column.

Ex: select min(marks) from student --------returns minimum value from marks column in student table.

4. In SQL, create a pivot query.

Pivot is used to transform one table into another in order to achieve more simpler view of table. Conventionally we can say that Pivot operator converts the rows data of the table into the column data.

Syntax:

1. Pivot:

SELECT (ColumnNames)

FROM (TableName)

PIVOT

(

AggregateFunction(ColumnToBeAggregated)

FOR PivotColumn IN (PivotColumnValues)

) AS

5. With an example, describe how to join in SQL.

A SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are:

• INNER JOIN

• LEFT JOIN

• RIGHT JOIN

• FULL JOIN

INNER JOIN: The INNER JOIN keyword selects all rows from both the tables as long as the condition satisfies. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be same.

\*/

use [H+Active]

select \* from

[dbo].[Invoices] I inner join [dbo].[Customers] C

on I.[CustomerID] = C.[CustomerID]

/\*

LEFT JOIN: This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of join. The rows for which there is no matching row on right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN.

\*/

select \* from [dbo].[Invoices] I left join [dbo].[Customers] C

on I.[CustomerID] = C.[CustomerID]

/\*

RIGHT JOIN: RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of join. The rows for which there is no matching row on left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.

\*/

select \* from [dbo].[Invoices] I right join [dbo].[Customers] C

on I.[CustomerID] = C.[CustomerID]

/\*

FULL JOIN: FULL JOIN creates the result-set by combining result of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both the tables. The rows for which there is no matching, the result-set will contain NULL values.

\*/

select \* from [dbo].[Invoices] I full join [dbo].[Customers] C

on I.[CustomerID] = C.[CustomerID]

/\*

cross JOIN: In a cross JOIN there is a join for each row of one table to every row of another table. This usually happens when the matching column or WHERE condition is not specified.

In the absence of a WHERE condition the cross JOIN will behave like a CARTESIAN PRODUCT . i.e., the number of rows in the result-set is the product of the number of rows of the two tables.

In the presence of WHERE condition this JOIN will function like a INNER JOIN.

\*/

select \* from [dbo].[Invoices] I cross join [dbo].[Customers] C

/\*

SELF JOIN: As the name signifies, in SELF JOIN a table is joined to itself. That is, each row of the table is joined with itself and all other rows depending on some conditions. In other words we can say that it is a join between two copies of the same table

\*/

--6. How to locate the 4th highest value in a column in a row. Create your table.

use [H+Active]

select \* from [dbo].[ProductListing]

where [Price] =

(

select MIN([Price]) from [dbo].[ProductListing]

where [Price] in

(select distinct top 4 [Price] from [dbo].[ProductListing]

order by [Price] desc)

)

1. For an online purchasing database, create entity relationship diagrams. Create a database object from your entity diagram. (Worldwide importer data base)

